Министерство образования

Российской Федерации

Владимирский политехнический колледж

#### Курсовой проект по предмету

“Объектно-Ориентированное программирование”

Пояснительная записка

КП ПО-399.16.02 ПЗ

# Отчет составил:

Рачков Р.В.

студент гр. ПО-399

10.04.02

Отчет принял:

преподаватель Серкин С.Г.

15.04.02

Принято с оценкой:

Владимир 2002г.

СОДЕРЖАНИЕ

Введение.

1.Общая часть.

1.1.Постановка задачи.

1.1.1.Название задачи.

1.1.2.Математическое описание задачи.

1.1.3.Требования предъявляемые модели.

1.2.Описание языка программирования.

1.2.1.Выбор языка программирования.

2.Специальная часть.

2.1.Описание алгоритмической модели.

2.1.1.Входные данные.

2.1.2.Выходные данные.

2.2. Целевое назначение процедур и функций.

2.3. Инструкция по выполнению программы.

2.4. Сообщения программы.

3. Список используемой литературы.

Заключение.

4. Приложение.

4.1.Схема алгоритма.

4.2.Текст программы.

**Введение**

В двадцать первом веке широкое распространение получили автоматизация и компьютеризация производства. Эти процессы облегчают использование и эксплуатацию различных технических и производственных данных. Компьютер облегчает поиск, хранение и эксплуатацию различной документации. Сейчас на каждом производстве можно встретить отделы вычислительных центров. В них специально обученные люди занимаются изучением и созданием новых программ, которые облегчают труд работникам производства. Компьютеры стали настолько внедрены в жизнь и деятельность человека, что используются не только на производстве, но и для обучения и развития человека. Многие обучающие программы носят игровой характер. Это дает возможность человеку не затрудняя себя приобрести те или иные навыки которые пригодятся ему в жизни.

В наше время большое распространение получили игры. Они служат не только для отдыха и развлечений, но и содержат в себе обучающие элементы. Игры с обучающими элементами могут развить логику ребенка. Это поможет ему в предстоящей ему жизни.

Чтобы создать компьютерную игру необходимо написать программу. Программа, работающая на компьютере, не редко отождествляется с самим компьютером, так как человек, использующий программу “вводит в компьютер” исходные данные с клавиатуры, и “компьютер выдает результат” на экран. На самом деле преобразование исходных данных, вводимых с клавиатуры или из файла, в результате выводится на экран монитора. Процессор выполняет преобразования в соответствии с последовательностью команд. Таким образом, чтобы компьютер выполнил некоторую работу, необходимо разработать последовательность команд, обеспечивающую выполнение этой работы, т.е. написать программу.

Смысл этого курсового проекта состоит в том, чтобы создать программу игрового типа “Клон”. При запуске файла игры появляется окно на котором расположено игровое поле представляет собой клеточное поле (сетку) десять на десять, в котором при нажатии на кнопку “Новая игра” в четырех углах появляются фишки в виде “мордашек”, две синих и две красных, которые располагаются крест на крест. В игре участвуют два игрока, каждый игрок должен делать по одному ходу. Суть игры состоит в том, чтобы один из игроков смог перекрасить как можно больше чужих фишек, либо в середине игры, либо по окончанию игры. Игра заканчивается после того, как на игровом поле не останется фишек одного цвета, либо все поле будет заполнено, тогда появляется окно со счетом красных и синих фишек. Также целью игрока является просчитование каждого хода для того, чтобы не дать возможность противнику перекрасить свои фишки, и тем самым победить в этой игре и набрать большее количество очков, чем его противник, если поле уже заполнено. Эта игра позволяет пользователю улучшить свои умственные способности, развить логическое мышление. Играя в эту игру он может показать свое превосходство над противником, тем самым доказать свою целеустремленность. Учит его быстрой реакции на изменяющиеся условия игры. Игра была полностью написана на языке Delphi 6. Delphi-это среда разработки программ, ориентированных на работу в Windows. В основе идеологии Delphi лежат технология визуального проектирования и методология объектно-ориентированного программирования. Для представления программы Delphi используется язык Object Pascal. Среда Delphi - это сложный механизм, обеспечивающий высокоэффективную работу программиста. Delphi - превосходный инструмент, с помощью которого программист может создать прекрасный интерфейс пользователя к прикладным программам самых различных классов. Кроме того, Delphi позволяет работать с любыми базами данных, создавать прикладные программы для работы с Интернет и многое другое. Delphi - постоянно развивающаяся система. Она дает пользователю большой спектр возможностей. Воспользовавшись ими, программист может создавать различного рода программы. Даже такие как игры.

**1.Общая часть.**

1.1.Постановка задачи.

1.1.1.Название задачи.

Необходимо создать программу игрового типа. Игровое поле представляет собой клеточное поле (сетку) десять на десять (Рис. 1), в котором при нажатии на кнопку “Новая игра” в четырех углах появляются фишки в виде “мордашек”, две синих и две красных, которые располагаются крест на крест.

Рис. 1 Игровое поле.

В игре участвуют два игрока, каждый игрок должен делать по одному ходу (по умолчанию в начале игры ходят синие фишки, начиная со второй, и все последующие игры ходы чередуются). Суть игры состоит в том, чтобы один из игроков смог перекрасить как можно больше чужих фишек, либо в середине игры, либо по окончанию игры. Каждый игрок может либо прыгнуть через одну ячейку, тем самым сделав свой ход, либо щелкнув в соседнею ячейку он увеличивает свои шансы на победу (вставляет в эту ячейку свою фишку). Игроки должны быть внимательней в своих действиях и просчитывать свои ходы до мелочей, в противном случае его соперник может воспользоваться его оплошностью. Прежде чем ходить фишка сначала выделяется, а затем игрок щелкает мышью по той ячейке в которую он хочет, либо копировать, либо шагнуть (Рис. 2).

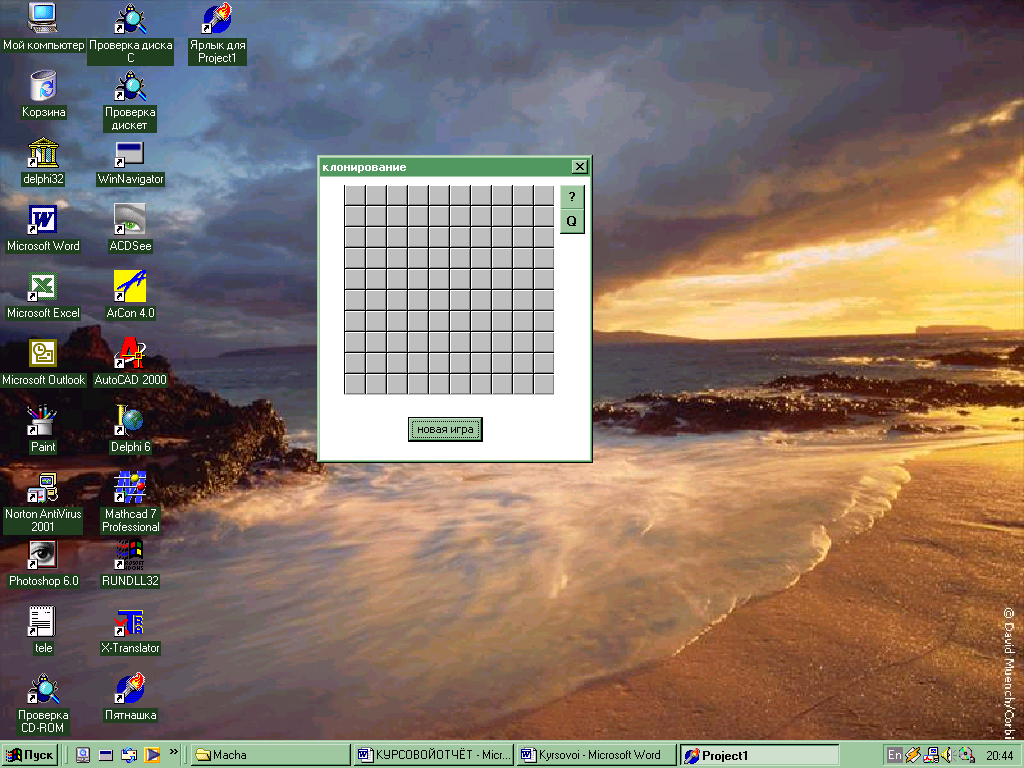
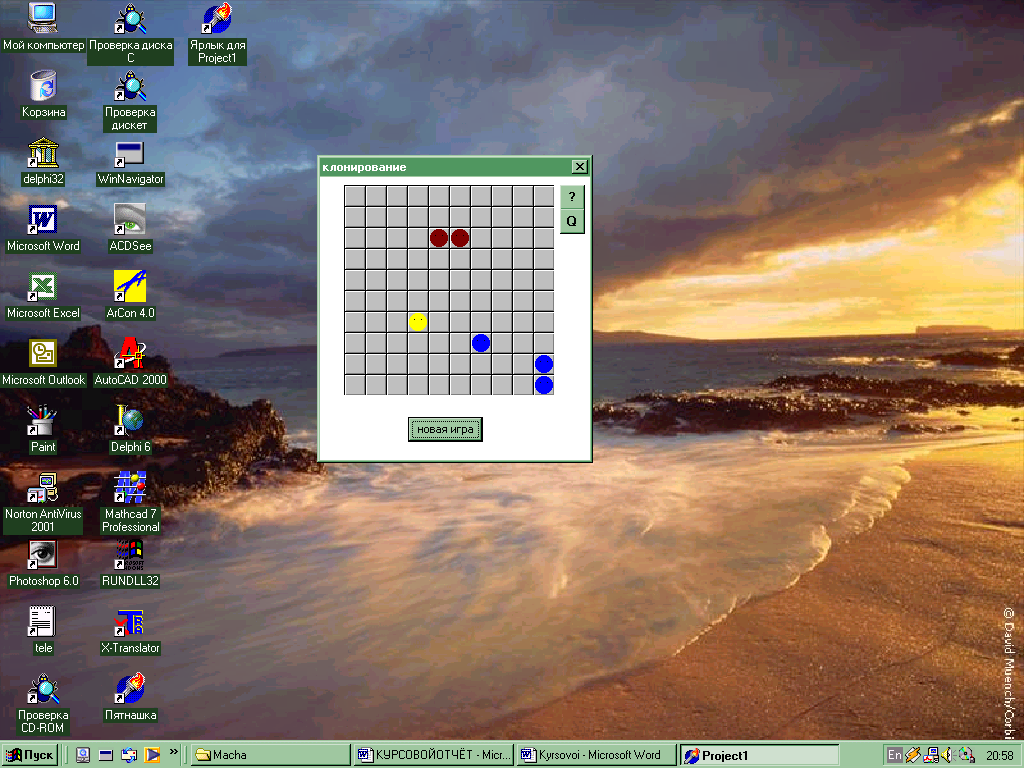


Рис. 2 Вид игры в начале.

Но у игрока есть шанс отменить свое выделение, если он передумал ходить этой фишкой, он просто должен щелкнуть другой клавишей мыши и выделение снимется. Программа игрового типа «Клон» предназначена для людей любого возраста. Как и все игры такого же типа «Клон» создана для развития логического мышления, интеллекта, умственных способностей и стратегических навыков человека, программа полезна для детей школьного возраста, а также для тех у кого много лишнего времени. Игра с хорошим интерфейсом и архитектурой будет не трудной в использовании.



Программа игрового типа «Клон» называется так, потому что на игровом поле происходит противоборство синих и красных “мордашек” в виде кружка с двумя глазками, и они постоянно друг друга перекрашивают, то синие, то красные, пока на поле не останется фишек только одного цвета или не останется места для «клонирования».

Для разработки игры мы использовали среду разработки Delphi 6, т.к. она имеет все ресурсы, методы и объекты для создания 32-х разрядных приложений, в том числе и игр. На наш взгляд для написания программ подобных нашей игре лучше всего использовать Object Pascal.

1.1.2.Математическое описание задачи.

В начале программы мы объявили глобальные переменные Vi, Vj, V2i, V2j типа byte, red и blue целочисленные. Потом идет описание двух массивов «a» и «b», «а» -целочисленный, который служит для записи чисел 0,1 и 2, определяющих пустую клетку, синие и красные фишки с порядковыми номерами i и j, которые параллельно применяются в массиве b типа TRECT, вследствие чего достигается параллельная работа двух массивов. Первый массив, предположим записывает 1 в ячейку а[i,j], а второй массив одновременно записывает в игровое поле рисунок соответствующий этой ячейке. Так как массив “b” полностью «натягивается» на поле и представляет собой квадрат или прямоугольник разбитый на кубики. К кубикам можно обращаться по координатам верхнего левого угла и нижнего правого угла кубика.

В нашей программе создано много циклов и условий, опишем только основные. В начале основной процедуры Formmousup описываем переменную HOD логического типа, которая служит для реализации по ходового режима игры. HOD в начале игры (при создании формы) истина. Для синих истина для красных лож. Таким образом процедура разбита на два блока: работа с синими и работа с красными. Поэтому описание математических действий программы можно сделать по первому блоку, т.к. второй идентичный.

Итак, описание первого блока. Проверим если HOD истина, то разрешается ходить синим (для красных лож), после этого начинается решение всего блока, который в свою очередь разбит на две части. Первая часть делает выделение выбранной фишки желтым цветом (у синих и красных одинаково). Проверим синяя ли эта фишка через массив, т.е. элемент a[i,j] должен быть равен 1, здесь же проверим левая ли клавиша нажата, если да, то через массив «b» копируется в поле желтая фишка и сохраняет координаты начальной фишки в Vi и Vj для дальнейшей обработки. Если нажата другая клавиша, то обратно синяя.

Вторая часть блока является основной, т.к. здесь ведутся все «боевые» действия: прыжки фишек, копирование фишек и перекрашивание чужих фишек в свои. Здесь же проверяется поле на наличие свободного места для копирования или прыжка, т.е. идет подсчет синих и красных фишек, которые так же используюся при выводе счета, они описаны глобально целочисленным типом.

Вторая часть блока начинается с проверки на наличие пустой ячейки, т.е. если a[i,j] равно нулю, то она пустая. Здесь же проверим левая ли клавиша нажата, если да, то приступаем к расчету расстояния на которое может ходить фишка. Проверяем выбрана ли ячейка (пустая), в которую мы ходим ходить на расстояние не дальше 3-х ячеек, если да, то мы фишку просто перенесем, если это расстояние равно 2-м ячейкам, т.е. первоначальная станет пустой (a[i,j] равно 0). А выбранная синей (a[i,j] равно 1). И в массивах записываются соответствующие значения, после чего Vi и Vj присваиваем 255-и, что означает, что не выбрана ни одна ячейка, т.е. завершен ход. Но если же расстояние меньше 3-х и не 2, то идет копирование в выбранную ячейку.

Итак, после этой основной задачи все поле будет проверяться на наличие свободного места, если его нет, то появляется окно, где написано, что нет места и выводится счет красных и синих фишек. Если же есть свободная ячейка, то идет обнуление результатов, HOD присваивается значение лож и ходить разрешается красным.

1.1.3.Требования предъявляемые модели.

Минимальные требования: любая система под управление Windows 95 и 16 Кб на диске (exe-файл 528 байт). Рекомендуемые требования представлены в таблице 1.

Таблица 1. Рекомендуемые требования.

|  |  |
| --- | --- |
| Параметры | Значение |
| Процессор | Intel Pentium 100 |
| Оперативная память | 8 Мб + память под ОС |
| ОС | Windows |
| Видео карта | Любая видео карта |

Программа может находиться на любом дисковом носителе, и будет занимать различное количество памяти (16 Кб).

Для компиляции программы необходим пакет Borland Delphi версии 3 и выше. Исходная программа компилировалась на Borland Delphi 6.

1.2.Описание языка программирования.

1.2.1.Выбор языка программирования.

Интерес к программированию постоянно растет. Это связано с развитием и внедрением в повседневную жизнь информационных технологий. Если человек имеет дело с компьютером, то рано или поздно у него возникает желание, а иногда и необходимость, научиться программировать.

Среди пользователей персональных компьютеров в настоящее время наиболее популярно семейство операционных систем Windows и, естественно, что тот, кто собирается программировать, стремится писать программы, которые будут работать в этих системах.

Несколько лет назад рядовому программисту оставалось только мечтать о создании собственных программ, работающих в среде Windows, т.к. единственным средством разработки был Borland C++ for Windows, явно ориентированный на профессионала, обладающих серьёзными знаниями и опытом.

Бурное развитие вычислительной техники, потребность в эффективных средствах разработки программного обеспечения привели к появлению систем программирования, ориентированных на так называемую «быструю разработку», среди которых можно выделить Borland Delphi и MS Visual Basic. В основе систем быстрой разработки (RAD-систем, Rapid Application Development-среда быстрой разработки приложений) лежит технология визуального проектирования и событийного программирования, суть которой заключается в том, что среда разработки берет на себя большую часть генерации кода программы, оставляя программисту работу по конструированию диалоговых окон и функций обработки событий. Производительность программиста при использовании RAD-систем – фантастическая!

Delphi – это среда быстрой разработки, в которой в качестве языка программирования используется Object Pascal. В основе идеологии Delphi лежит технология визуального проектирования и методология объектно-ориентированного событийного программирования.

Для написания программы игрового типа “Клон” использована среда программирования Delphi. Не секрет, что лучшим языком для изучения и освоения программирования является Паскаль, а лучшей в мире системой программирования для MS-DOS – Turbo Pascal. Среда программирования Delphi продолжила серию Паскаль-ориентированных средств программирования и, по моему глубокому убеждению, является наиболее удобным инструментом для Windows-программирования.

Среда программирования Delphi – это сложный механизм, обеспечивающий высокоэффективную работу программиста.

Эволюция технических средств персональных компьютеров привела к повсеместному вытеснению старой “доброй” ОС MS-DOS значительно более мощными системами Windows, программирование для которых существенно сложнее, чем программирование для MS-DOS. Разработчики систем программирования, и прежде всего такие корпорации, как Microsoft и Borland, не замедлили выпустить соответствующие средства: уже в 1991 году, т. е. сразу после появления Windows 3.1, Borland выпускает Turbo Pascal for Windows, а в 1992 – усовершенствованную версию этой системы программирования - Borland Pascal with Objects 7.0. Эти первые специализированные инструменты требовали основательного знания Windows и были сложны в освоении. Но вот в 1993 году Microsoft выпустила первую визуальную среду программирования Visual Basic, и программирование для Windows стало даже проще, чем программирование для MS-DOS. В ответ на это Borland в 1995 году выпустила первую версию Delphi, а затем, с интервалом в 1 год, - ещё 4 версии: 2, 3, 4, и 5. Наконец, в середине 2001года выпускается версия 6, т. е. почти через 2 года после выхода 5-й версии. Эту задержку можно объяснить тем, что параллельно с работой над новой версией Delphi шла разработка варианта Delphi для ОС Linux – эта система программирования вышла в феврале 2001 года и получила название Kylix. В результате 6-я версия Delphi имеет уникальную особенность: она способна создавать так называемые межплатформенные приложения, т. е. программы, которые одинаково успешно могут работать как под управлением Windows 32, так и под Linux.

До появления Kylix единственными средствами программирования для Linux были системы на языках C/C++, Basic, Fortran. Kylix, а следом за ней – Delphi 6, открываются богатейшие возможности VCL (вариант которой под Linux/Windows называется CLX – Borland Component Library for Cross-Platform Application) для разработки прикладных программ для Linux, что позволяет говорить об этой ОС как о потенциальном и весьма опасном конкуренте Windows.

Две другие особенности Delphi 6 также достойны особого упоминания: в ней сделаны дальнейшие шаги для поддержки Web-программирования (архитектура *WebSnap*) и разработаны драйверы и компоненты для максимально быстрой связи клиентских мест с некоторыми популярными промышленными серверами баз данных без BDE (компоненты страницы *dbExpress*).

Таким образом, основой идей Delphi 6 является обеспечение перехода от дорогих патентованных решений корпорацииMicrosoft к бесплатным (или почти бесплатным) решениям на базе Linux.

**2.Специальная часть.**

2.1.Описание алгоритмической модели.

2.1.1.Входные данные.

Данная программа игрового типа “Клон” не требует никаких входных данных. Игрок просто запускает \*.exe-файл и сразу перед ним появляется окно на котором располагается игровое поле. Для начала игры нужно нажать кнопку ”Новая игра”. Чтобы завершить игру нужно нажать кнопку “Q” или нажатием мышкой на крестик окна.

2.1.2 Выходные данные.

Выходными данными является окно “Результат” (Рис. 3), которое появляется при заполнении всего поля фишками, как на рисунке представленном ниже.

Рис. 3 Окно “Результат”

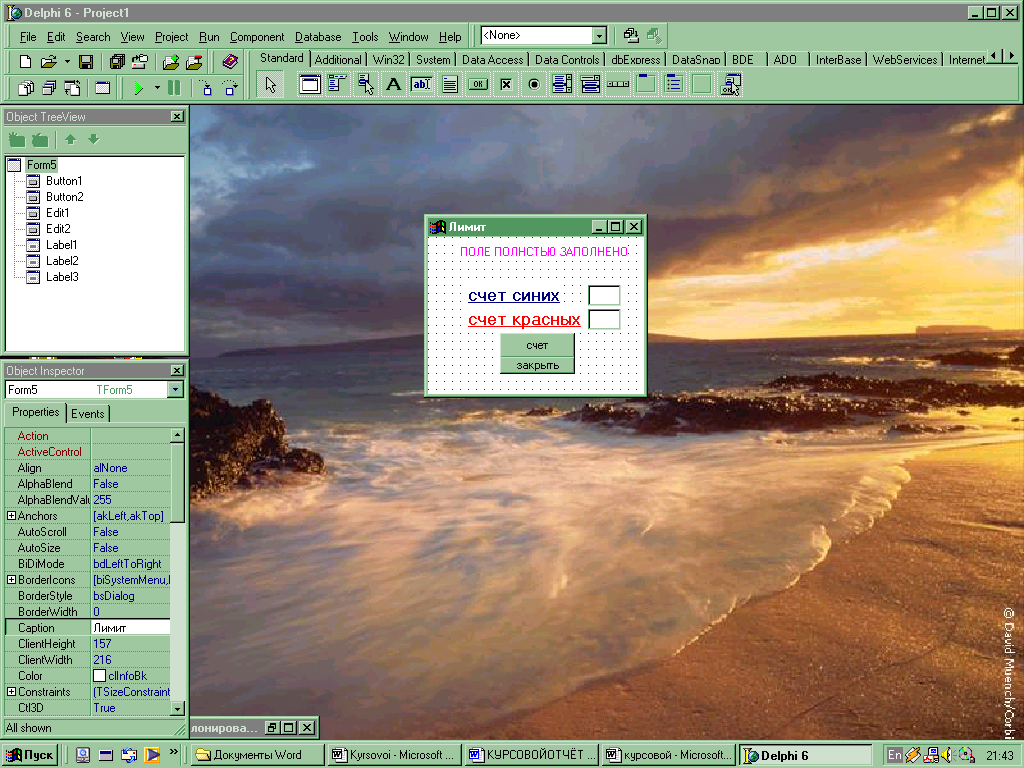
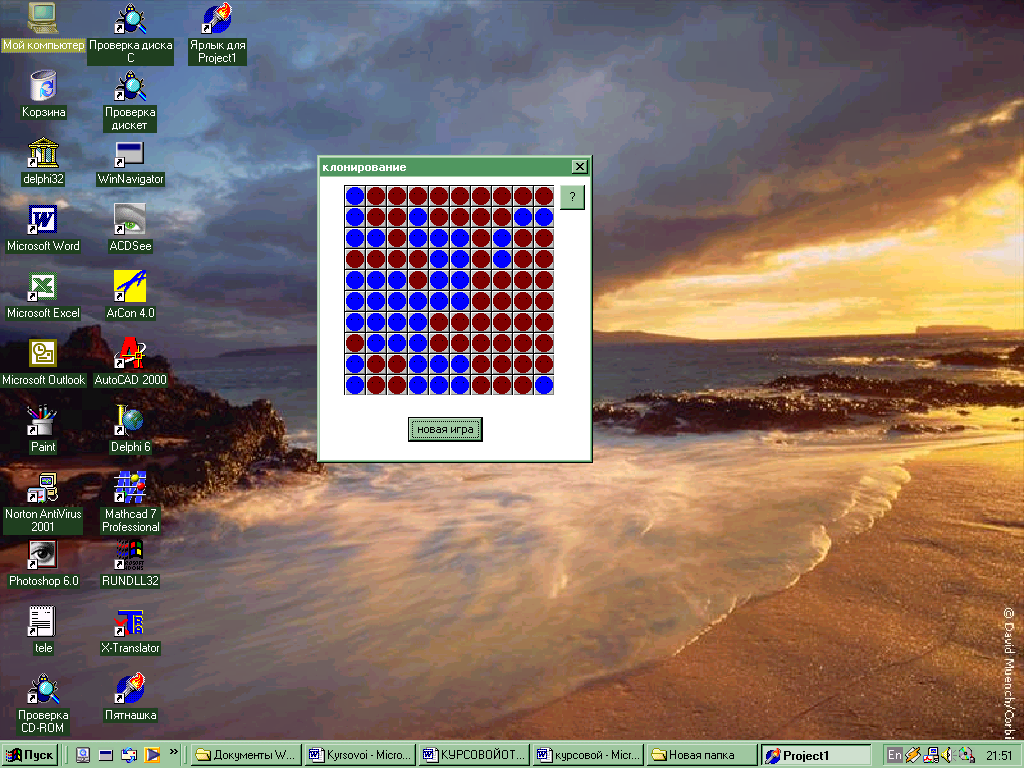


Рис. 4 Полностью забитое поле.



2.2 Целевое назначение процедур и функций.

|  |  |  |
| --- | --- | --- |
| Название процедуры | Параметры  процедуры | Назначение процедуры |
| formmouseup | (Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer) | Процедура выполняет большинство всех операций, проверок и условий во всей программе, она является основополагающей всей программы, выполняет проверку нажатий левой и правой клавишей мыши, в соответствии с чем выполняет те или иные операции, а в частности копирование нужных рисунков (\*.bmp) в нужную ячейку, после установки нужного рисунка выполняет проверку близлежащих ячеек около установленного рисунка, а также проверяет всё поле на наличие той или иной ситуации на всём игровом поле. |
| TForm1.Button1Click | (Sender: TObject) | описывает кнопку «Новая игра», при нажатии на которую, все поле заполняется пустыми ячейками (пустыми элементами поля) и по углам устанавливаются 2-синих и 2-красных фишки, относительно друг друга стоят по диагонали. |
| TForm1.FormCreate | (Sender: TObject) | присваивает элементам массива (b:byte) координаты ячеек поля и адрес ячейки в массиве сопоставлен ячейке на поле.  procedure TForm1.Button2Click(Sender: TObject) – при нажатии на кнопку [?] выводится информация о программе на Form2. |
| TForm1.Button2Click | (Sender: TObject) | при нажатии на кнопку [?] выводится информация о программе на Form2. |
| TForm2.Button1Click | (Sender: TObject) | на Form2 расположена кнопка [понял], которая закрывает окно справки о программе. |
| TForm3.BitBtn1Click | (Sender: TObject) | описывает графическую кнопку [победа] с рисунком , которая закрывает Form3. |
| TForm4.BitBtn1Click | (Sender: TObject) | описывает графическую кнопку [победа] с рисунком, которая закрывает Form4. |
| TForm5.Button1Click | (Sender: TObject) | описывает кнопку, которая закрывает форму сообщения о том, что поле полностью заполнено (Form5). |

Во всех выше перечисленных процедурах используется параметр Sender, который передается во все обработчики событий компонентов Delphi как объект – источник события и имеющий тип TObject.

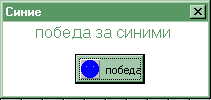
2.3 Инструкция по выполнению программы.

Для запуска программы необходимо найти и двойным щелчком мыши нажать на exe-файл. После запуска программы появится окно на котором располагаются клеточное поле десять на десять, кнопка “Новая игра”, кнопка ”?” - помощь и кнопка “Q” – выход. Для того чтобы начать игру необходимо нажать на кнопку “Новая игра”, после чего на поле в четырех углах появятся фишки две красных и две синих друг против друга, крест на крест. Передвижение фишек по полю осуществляется путем нажатия на левую и правую кнопки мыши. Синие фишки передвигаются левой кнопкой, а красные правой кнопкой мыши. Прежде чем ходить, фишка выделяется желтым цветом, это значит, что она сейчас ходит, а отменить выделение можно путем нажатия на противоположную кнопку мыши. Игра “Клон” означает клонирование или размножение фишек. Фишки могут ходить лишь в пределах двух ячеек, в соседнею ячейку они копируются, а через ячейку они прыгают. Перекрашивание происходит так, если в пределах двух ячеек вокруг фишки стоят чужие, то они перекрашиваются. Игра закончится тогда, когда на поле останутся фишки одного цвета или все поле будет заполнено и тогда выдается счет красных и синих фишек. Кнопка ”?” при нажатии на которую появляется окно в котором вкратце описано в чем заключается игра. Кнопка “Q” закрывает игровое окно.

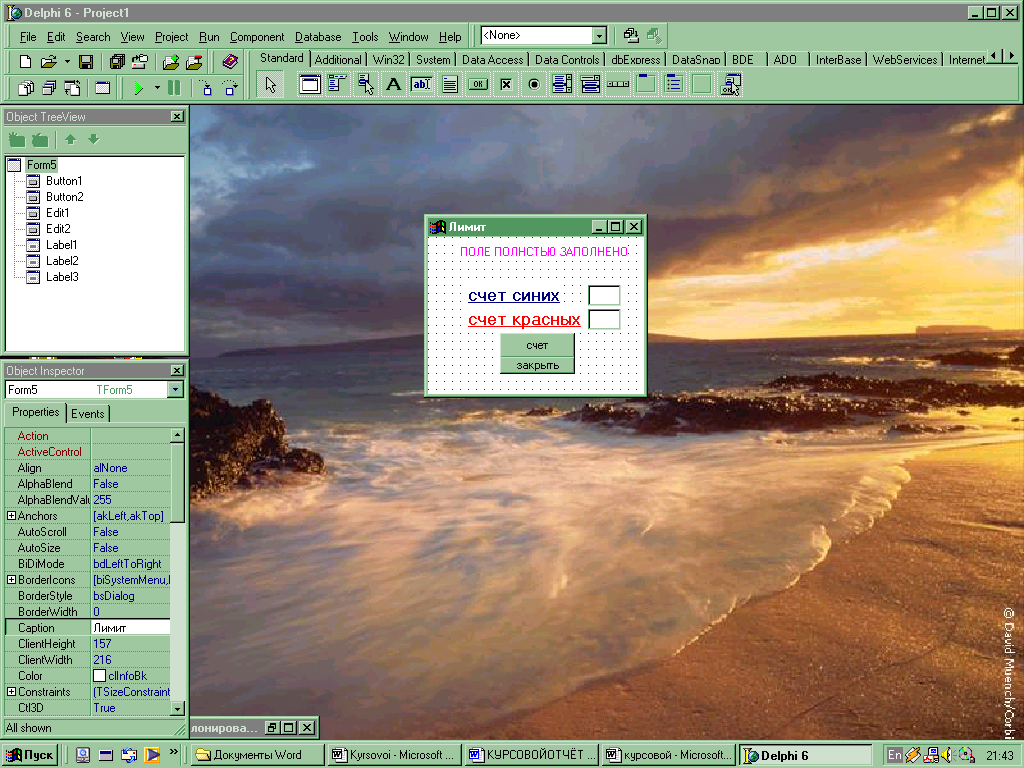
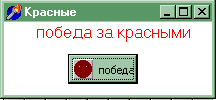
2.4 Сообщения программы.

В процессе работы программы возможна активизация окон-сообщений:

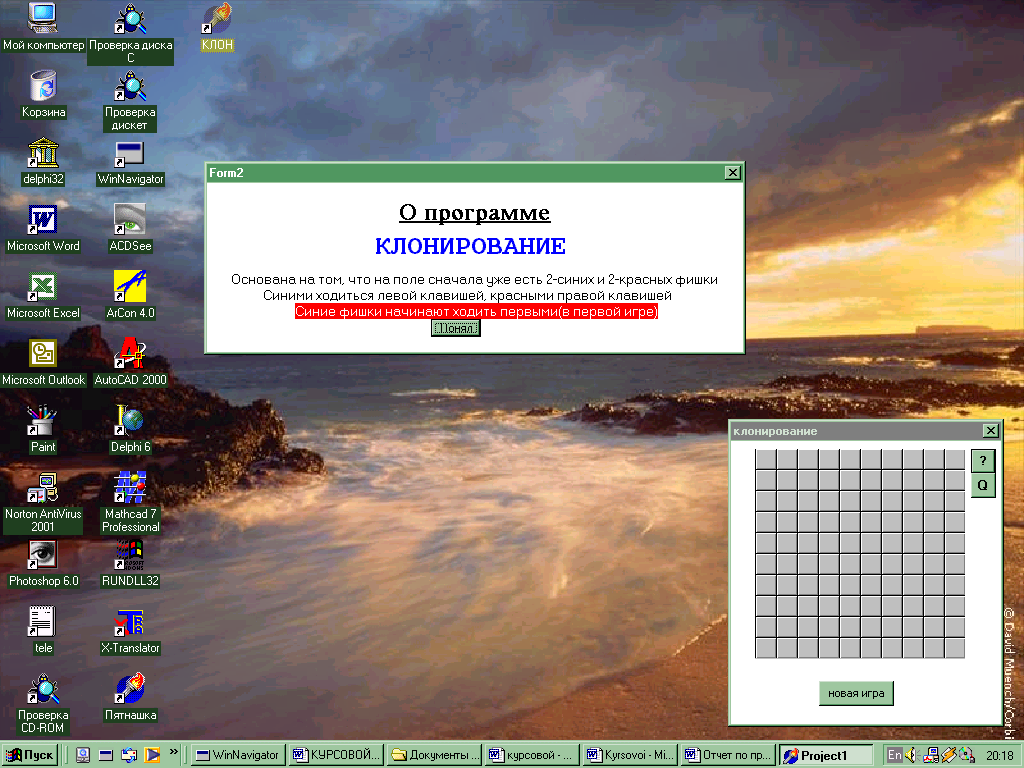
1. Данное диалоговое окно активизируется, когда синие фишки произвели «перекрашивание» всех красных фишек, все операции необходимые для этого описаны



2. Данное диалоговое окно активизируется, когда красные фишки произвели «перекрашивание» всех синих фишек, все операции необходимые для этого описаны



3. Данное диалоговое окно активизируется, когда все поле заполнено, и некуда ходить, то выводится счет красных и синих фишек



4. Окно активизируется при нажатии на кнопку [?] – о программе

**3. Список использованной литературы.**

1. Архангельский А. Я. Delphi 5: “Справочное пособие”;
2. Фаронов В. В. “Delphi 5: учебный курс” М: Издательство ”Нолидж”, 1999;
3. В.В. Фаронов «Delphi 6»;

4. Н. Культин «Программирование в Turbo Pascal 7.0 и Delphi »;

5. Н. Культин «Delphi. Программирование на Object Pascal» - издательство ”БВХ-Петербург”, 2001.

**Заключение.**

Во время создания игры “Клон” мы познали дополнительные возможности Delphi. Несмотря на наши сомнения, программа получилась оригинальной, с хорошим пользовательским интерфейсом. Конечно, она может дорабатываться и улучшаться. Эта игра в будущем может быть очень популярна. Мы подумываем над сетевыми возможностями этой игры, в которую смогут играть в обеденный перерыв бухгалтеры разных фирм.

Таким образом, программирование с использованием среды Delphi, позволило нам составить игру “Клон” довольно просто и быстро. Игра получилась достаточно компактной, не требовательная к ресурсам машины, очень проста в обращении и развивающая умственные способности человека.

Среда программирования Delphi – это сложный механизм, обеспечивающий высокоэффективную работу программиста.

**Приложение.**

4.1 Схема алгоритма.

4.2 Текст программы.

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,

Grids, ExtCtrls, Buttons, Menus, StdCtrls;

type

TForm1 = class(TForm)

Image1: TImage;

image3: TImage;

Image2: TImage;

Button1: TButton;

Image4: TImage;

Image5: TImage;

Button2: TButton;

Button3: TButton;

procedure formmouseup(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

a: array [0..9,0..9] of byte;

b: array [0..9,0..9] of trect;

vi,vj,v2i,v2j:byte;

red,blue:integer;

hod:boolean;

implementation

uses Unit2, Unit3, Unit4, Unit5;

{$R \*.DFM}

procedure TForm1.formmouseup(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

var

i,j:byte;

ai,aj,a2i,a2j:integer;

kol,m,n:integer;

begin

blue:=0;

red:=0;

kol:=0;

i:=x div (image1.height);

j:=y div (image1.height);

//-------------------------------------------------------------------------------

//При нажатии активная синяя фишка (левой клавишей)

if hod=true then

BEGIN

if (a[i,j]=1) then

begin

if button=mbLeft then

begin

image2.Canvas.StretchDraw(b[i,j],image4.picture.bitmap);

Vi:=i;

Vj:=j;

end

else

begin

image2.Canvas.StretchDraw(b[i,j],image1.picture.bitmap);

end;

end;

//-----------------------------------------------------------------------------

//Возвращение рисунка для синих фишек

if (a[i,j]=0) then

if button=mbLeft then

begin

if (abs(i-vi)<3) and (abs(j-vj)<3) then

begin

if (abs(i-vi)=2) or (abs(j-vj)=2) then

begin

a[vi,vj]:=0;

image2.Canvas.StretchDraw(b[vi,vj],image5.picture.bitmap);

image2.Canvas.StretchDraw(b[i,j],image1.picture.bitmap);

a[i,j]:=1; vi:=255;vj:=255;

end

else

begin

image2.Canvas.StretchDraw(b[i,j],image1.picture.bitmap);

if (vi<>255)and (vj<>255) then

begin

image2.Canvas.StretchDraw(b[vi,vj],image1.picture.bitmap);

vi:=255; vj:=255;

end;

a[i,j]:=1;

end;

for ai:=i-1 to i+1 do //проверка близлежащих фишек КРАСНЫХ

for aj:=j-1 to j+1 do

if (ai>=0) and (aj>=0)and (ai<=9) and(aj<=9) then

if a[ai,aj]=2 then

begin

a[ai,aj]:=1;

image2.Canvas.StretchDraw(b[ai,aj],image1.picture.bitmap);

beep;

for m:=i-9 to i+9 do //проверка всего поля

for n:=j-9 to j+9 do

if (m>=0)and(n>=0)and(m<=9)and(n<=9) then

begin

if (a[m,n]=1)or(a[m,n]=0)then

kol:=kol+1;

end;

if kol=100 then

form4.show;

m:=0;n:=0;kol:=0;

end;hod:=false;//теперь красные

end;

end;

END;

//-------------------------------------------------------------------------------

//При нажатии активна красная (правая клавиша)

if hod=false then

BEGIN

if a[i,j]=2 then

begin

if button=mbright then

begin

image2.Canvas.StretchDraw(b[i,j],image4.picture.bitmap);

V2i:=i;

v2j:=j;

end

else

begin

image2.Canvas.StretchDraw(b[v2i,v2j],image3.picture.bitmap);

end;

end;

//-----------------------------------------------------------------------------

//Возвращение рисунка для красных фишек

if (a[i,j]=0) then

if button=mbright then

begin

if (abs(i-v2i)<3) and (abs(j-v2j)<3) then

begin

if(abs(i-v2i)=2) or (abs(j-v2j)=2) then

begin

a[v2i,v2j]:=0;

image2.Canvas.StretchDraw(b[v2i,v2j],image5.picture.bitmap);

image2.Canvas.StretchDraw(b[i,j],image3.picture.bitmap);

a[i,j]:=2; v2i:=255; v2j:=255;

end

else

begin

image2.Canvas.StretchDraw(b[i,j],image3.picture.bitmap);

if (v2i<>255)and(v2j<>255) then

begin

image2.Canvas.StretchDraw(b[v2i,v2j],image3.picture.bitmap);

v2i:=255; v2j:=255;

end;

end;

a[i,j]:=2;

for a2i:=i-1 to i+1 do //проверка близлежащих фишек СИНИХ

for a2j:=j-1 to j+1 do

if (a2i>=0)and(a2j>=0)and(a2i<=9)and(a2j<=9) then

if a[a2i,a2j]=1 then

begin

a[a2i,a2j]:=2;

image2.Canvas.StretchDraw(b[a2i,a2j],image3.picture.bitmap);

beep;

end;

for m:=i-9 to i+9 do //проверка всего поля

for n:=j-9 to j+9 do

if (m>=0)and(n>=0)and(m<=9)and(n<=9) then

begin

if (a[m,n]=2)or(a[m,n]=0)then

kol:=kol+1;

end;

if kol=100 then

form3.show;

hod:=true;//теперь синие

end;

end;

END;

m:=0;n:=0;kol:=0;

//-----------------------------------------------------------------------------

//Прверка поля на наличие свободного места

for m:=i-9 to i+9 do //проверка всего поля

for n:=j-9 to j+9 do

if (m>=0)and(n>=0)and(m<=9)and(n<=9) then

begin

if (a[m,n]=1) then blue:=blue+1; //счет синих

if (a[m,n]=2) then red:=red+1; //счет красных

kol:=red+blue;

end;

if kol=100 then

form5.show;

m:=0;n:=0;kol:=0;

end;

//-----------------------------------------------------------------------------

procedure TForm1.Button1Click(Sender: TObject);

var i,j:byte;

begin

for i:=0 to 9 do

for j:=0 to 9 do

begin

image2.Canvas.StretchDraw(b[i,j],image5.picture.bitmap);

a[i,j]:=0;

end;

begin

image2.Canvas.StretchDraw(b[0,0],image1.picture.bitmap);

a[0,0]:=1;

image2.Canvas.StretchDraw(b[0,9],image3.picture.bitmap);

a[0,9]:=2;

image2.Canvas.StretchDraw(b[9,9],image1.picture.bitmap);

a[9,9]:=1;

image2.Canvas.StretchDraw(b[9,0],image3.picture.bitmap);

a[9,0]:=2;

end;

end;

procedure TForm1.FormCreate(Sender: TObject);

VAR I,J:BYTE;

begin

hod:=true;

for i:=0 to 9 do

for j:=0 to 9 do

begin

b[i,j].Left:=i\*(image1.Width);

b[i,j].top:=j\*(image1.Height);

b[i,j].Bottom:=b[i,j].Top+image1.Height;

b[i,j].Right:=b[i,j].Left+image1.Width

end;

end;

procedure TForm1.Button2Click(Sender: TObject);

begin

form2.show;

end;

procedure TForm1.Button3Click(Sender: TObject);

begin

close;

end;

end.

unit Unit2;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls;

type

TForm2 = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Button1: TButton;

Label5: TLabel;

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

{$R \*.dfm}

procedure TForm2.Button1Click(Sender: TObject);

begin

close;

end;

end.

unit Unit3;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Buttons;

type

TForm3 = class(TForm)

Label1: TLabel;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form3: TForm3;

implementation

{$R \*.dfm}

procedure TForm3.BitBtn1Click(Sender: TObject);

begin

close;

end;

end.

unit Unit4;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Buttons;

type

TForm4 = class(TForm)

BitBtn1: TBitBtn;

Label1: TLabel;

procedure BitBtn1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

procedure TForm4.BitBtn1Click(Sender: TObject);

begin

close;

end;

end.

unit Unit5;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Grids;

type

TForm5 = class(TForm)

Label1: TLabel;

Button1: TButton;

Edit2: TEdit;

Label2: TLabel;

Label3: TLabel;

Edit1: TEdit;

Button2: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

implementation

uses unit1;

{$R \*.dfm}

procedure TForm5.Button1Click(Sender: TObject);

begin

edit1.text:=inttostr(blue);

edit2.text:=inttostr(red);

end;

procedure TForm5.Button2Click(Sender: TObject);

begin

close;

end;

end.